Présentation des Données du Web - TP 3 : XML Relationnel

Abadie Martin - Bressand Jérémy

1) Oracle-XML : CLOB vs Binary XML

Requête : //article [@auteur = //journaliste[child::anonyme]/@idJ]
Temps d'exécution :
CLOB : Elapsed: 00:00:00.08
Binary XML : Elapsed: 00:00:00.45

Requête : //article [position() = 1]/corps
Temps d'exécution :
CLOB : Elapsed: 00:00:00.01
Binary XML : Elapsed: 00:00:00.01

Requête : //journaliste[nom]
Temps d'exécution :
CLOB : Elapsed: 00:00:00.01
Binary XML : Elapsed: 00:00:00.15

Requête : //description
Temps d'exécution :
CLOB : Elapsed: 00:00:00.03
Binary XML Elapsed: 00:00:00.00

Requête : //etage[descendant::nombrePlaces[text()=40]]/description
Temps d'exécution :
CLOB Elapsed: 00:00:00.01
Binary XML Elapsed: 00:00:00.05

Requête : //etage[count(descendant::salle)=3]/description
Temps d'exécution :
CLOB Elapsed: 00:00:00.01
Binary XML Elapsed: 00:00:00.06

Conclusion : On remarque que le requêtes sur les colonnes au format CLOB sont globalement plus rapides que les requêtes sur les colonnes au format Binary XML. Cependant, nos données XML sont de très petites taille, donc non représentatives, car dans un cas d'utilisation normal, le format Binary XML offre de meilleurs performances de réponse aux requêtes que le format CLOB.

2) Stockage schema-unaware : EDGE vs Vertical-EDGE vs Monet (sur les batiments)

1. Implémentation des tables SQL

creationEdge.sql :

```
create table edge (
        source varchar(5),
        target varchar(5),
        ordinal number,
        tag varchar(20),
        type varchar(3) NOT NULL,
        constraint pk_edge primary key (target),
        constraint fk_edge_edge foreign key(source) references edge(target)
);

create table textvalues (
        node varchar(5),
        value varchar(50),
        constraint pk_textvalues primary key (node),
        constraint fk_textvalues_edge foreign key (node) references edge(target)
);

create table numvalues (
        node varchar(4),
        value NUMBER,
        constraint pk_numvalues primary key (node),
        constraint fk_numvalues_edge foreign key (node) references edge(target)
);
```

creationVerticalEdge.sql :

```
create table batiment (
        source varchar(5),
        target varchar(5),
        ordinal number,
        txtval varchar(1),
        numval number,
        constraint pk_batiment primary key (target)
);


create table etage (
        source varchar(5),
        target varchar(5),
        ordinal number,
        txtval varchar(1),
        numval number,
        constraint pk_etage primary key (target),
        constraint fk_etage_batiment foreign key (source) references batiment(target)
);


create table description (
        source varchar(5),
        target varchar(5),
        ordinal number,
        txtval varchar(50),
        numval number,
        constraint pk_description primary key (target),
        constraint fk_description_etage foreign key (source) references etage(target)
);


create table salle (
        source varchar(5),
        target varchar(5),
        ordinal number,
        txtval varchar(1),
        numval number,
        constraint pk_salle primary key (target),
        constraint fk_salle_etage foreign key (source) references etage(target)
);

create table nombrePlaces (
        source varchar(5),
        target varchar(5),
        ordinal number,
        txtval varchar(1),
        numval number,
        constraint pk_nombrePlaces primary key (target),
        constraint fk_nombrePlaces_salle foreign key (source) references salle(target)
);
```

creationMonet.sql :

```
create table batiment_m (
        node varchar(5),
        textval varchar(1),
        numval number,
        constraint pk_batiment_m primary key (node)
);
create table batiment_etage (
        node varchar(5),
        textval varchar(1),
        numval number,
        constraint pk_batiment_etage primary key (node)
);
create table batiment_etage_description (
        node varchar(5),
        textval varchar(50),
        numval number,
        constraint pk_batiment_etage_description primary key (node)
);
create table batiment_etage_salle (
        node varchar(5),
        textval varchar(1),
        numval number,
        constraint pk_batiment_etage_salle primary key (node)
);
create table batiment_etage_salle_nbPlaces (
        node varchar(5),
        textval varchar(1),
        numval number,
        constraint pk_batiment_etage_nbPlaces primary key (node)
);
```

2. Remplissage des tables SQL

insertionEdge.sql :

```
insert into edge (source, target, ordinal, tag, type) values ( NULL, 'n0', NULL, 'batiment', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n0', 'n1', 1, 'etage', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n0', 'n2', 2, 'etage', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n1', 'n3', 1, 'description', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n1', 'n4', 2, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n1', 'n5', 3, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n1', 'n6', 4, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n2', 'n7', 1, 'description', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n2', 'n8', 2, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n2', 'n9', 3, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n2', 'n10', 4, 'salle', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n3', 'n11', 1, NULL, 'txt');
insert into edge (source, target, ordinal, tag, type) values ( 'n4', 'n12', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n5', 'n13', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n6', 'n14', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n7', 'n15', 1, NULL, 'txt');
insert into edge (source, target, ordinal, tag, type) values ( 'n8', 'n16', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n9', 'n17', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n10', 'n18', 1, 'nbPlaces', 'elt');
insert into edge (source, target, ordinal, tag, type) values ( 'n12', 'n19', 1, NULL, 'num');
insert into edge (source, target, ordinal, tag, type) values ( 'n13', 'n20', 1, NULL, 'num');
insert into edge (source, target, ordinal, tag, type) values ( 'n14', 'n21', 1, NULL, 'num');
insert into edge (source, target, ordinal, tag, type) values ( 'n16', 'n22', 1, NULL, 'num');
insert into edge (source, target, ordinal, tag, type) values ( 'n17', 'n23', 1, NULL, 'num');
insert into edge (source, target, ordinal, tag, type) values ( 'n18', 'n24', 1, NULL, 'num');


insert into textvalues (node, value) values ('n11', 'Rez-de-chaussee');
insert into textvalues (node, value) values ('n15', 'Premier etage');

insert into numvalues (node, value) values ('n19', 40);
insert into numvalues (node, value) values ('n20', 40);
insert into numvalues (node, value) values ('n21', 40);
insert into numvalues (node, value) values ('n22', 50);
insert into numvalues (node, value) values ('n23', 30);
insert into numvalues (node, value) values ('n24', 25);
```

insertionVerticalEdge.sql :

```
insert into batiment values (NULL, 'n0', NULL, NULL, NULL);

insert into etage values ('n0', 'n1', 1, NULL, NULL);
insert into etage values ('n0', 'n2', 2, NULL, NULL);

insert into description values ('n1', 'n3', 1, 'Rez-de-chaussee', NULL);
insert into description values ('n2', 'n7', 1, 'Premier etage', NULL);

insert into salle values ('n1', 'n4', 2, NULL, NULL);
insert into salle values ('n1', 'n5', 3, NULL, NULL);
insert into salle values ('n1', 'n6', 4, NULL, NULL);
insert into salle values ('n2', 'n8', 2, NULL, NULL);
insert into salle values ('n2', 'n9', 3, NULL, NULL);
insert into salle values ('n2', 'n10', 4, NULL, NULL);

insert into nombrePlaces values ('n4', 'n11', 1, NULL, 40);
insert into nombrePlaces values ('n5', 'n12', 1, NULL, 40);
insert into nombrePlaces values ('n6', 'n13', 1, NULL, 40);
insert into nombrePlaces values ('n8', 'n14', 1, NULL, 50);
insert into nombrePlaces values ('n9', 'n15', 1, NULL, 30);
insert into nombrePlaces values ('n10', 'n16', 1, NULL, 25);
```

insertionMonet.sql :

```
insert into batiment_m values ('n0', NULL, NULL);

insert into batiment_etage values ('n1', NULL, NULL);
insert into batiment_etage values ('n2', NULL, NULL);


insert into batiment_etage_description values ('n3', 'Rez-de-chaussee', NULL);
insert into batiment_etage_description values ('n7', 'Premier etage', NULL);


insert into batiment_etage_salle values ('n4', NULL, NULL);
insert into batiment_etage_salle values ('n5', NULL, NULL);
insert into batiment_etage_salle values ('n6', NULL, NULL);
insert into batiment_etage_salle values ('n8', NULL, NULL);
insert into batiment_etage_salle values ('n9', NULL, NULL);
insert into batiment_etage_salle values ('n10', NULL, NULL);
insert into batiment_etage_salle_nbPlaces values ('n11', NULL, 40);
insert into batiment_etage_salle_nbPlaces values ('n12', NULL, 40);
insert into batiment_etage_salle_nbPlaces values ('n13', NULL, 40);
insert into batiment_etage_salle_nbPlaces values ('n14', NULL, 50);
insert into batiment_etage_salle_nbPlaces values ('n15', NULL, 30);
insert into batiment_etage_salle_nbPlaces values ('n16', NULL, 25);
```

3. Requetes SQL

requetesEdge.sql :

```
--Requete : //description
select value
from textvalues t, edge e1, edge e2
where t.node = e1.target
and e1.source = e2.target
and e2.tag = 'description';

--Requête : //etage[descendant::nombrePlaces[text()=40]]/description
select distinct t.value
from textvalues t,
edge e1,
edge e2,
edge e3,
edge e4,
edge e5,
edge e6,
numvalues n
where n.value = 40
and n.node = e1.target
and e1.type = 'num'
and e1.source = e2.target
and e2.tag = 'nbPlaces'
and e2.source = e3.target
and e3.tag = 'salle'
and e3.source = e4.target
and e4.tag = 'etage'
and e5.source = e4.target
and e5.tag='description'
and e6.source = e5.target
and e6.type = 'txt'
and t.node = e6.target;




--Requête : //etage[count(descendant::salle)=3]/description
```

```
select t.value
from textvalues t,
edge e1,
edge e2
where t.node = e1.target
and e1.source = e2.target
and e2.tag = 'description'
and e2.source in (
        select etage from (
        select e3.target as etage, count(*)
        from edge e3, edge e4
        where e3.tag = 'etage'
        and e4.source = e3.target
        and e4.tag = 'salle'
        group by e3.target
        having count(*) = 3
));
```

requetesVerticalEdge.sql :

```
--Requete : //description
select txtval
from description;

--Requête : //etage[descendant::nombrePlaces[text()=40]]/description
select d.txtval
from description d,
etage e
where d.source = e.target
and e.target in (
        select e1.target
        from etage e1,
        salle s,
        nombrePlaces n
        where n.numval = 40
        and n.source = s.target
        and s.source = e1.target
);

--Requête : //etage[count(descendant::salle)=3]/description
select d.txtval
from description d,
etage e
where d.source = e.target
and e.target in (
        select etage from (
        select e1.target as etage, count(*)
        from etage e1, salle s
        where s.source = e1.target
        group by e1.target
        having count(*) = 3
));
```

requetesMonet.sql :

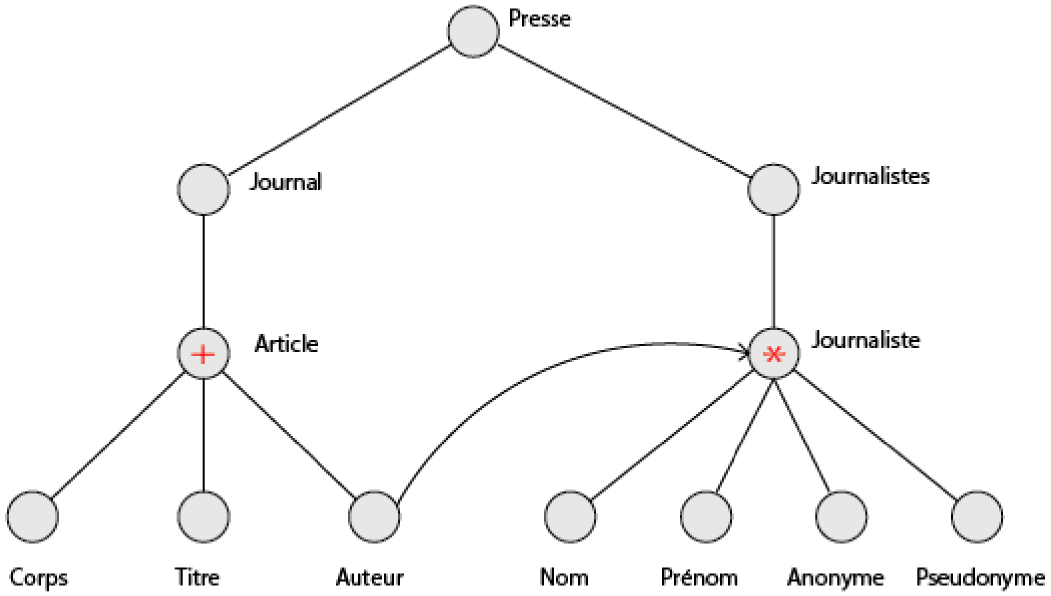
```
--Requete : //description
select textval
from batiment_etage_description;

--Requête : //etage[descendant::nombrePlaces[text()=40]]/description
--Impossible : on ne peut pas acceder au descendant d’un noeud

--Requête : //etage[count(descendant::salle)=3]/description
--Impossible : pareil que précedemment
```

3) Stockage schema-aware (sur la presse)

1.



2.

```
drop table Article;
drop table Journaliste;
drop table Presse;

create table Presse (
	presseID number,
	journal varchar(25),
	constraint pk_presse primary key (presseID)
);


create table Journaliste (
	journalisteID number,
	nom varchar(30),
	prenom varchar(30),
	pseudonyme varchar(30),
	anonyme varchar(1),
	presseID number,
	constraint pk_journaliste primary key (journalisteID),
    constraint fk_journaliste_presse foreign key (presseID) references
Presse(presseID)
);
```

```
create table Article (
       articleID number,
       corps varchar(1024),
       titre varchar(50),
       presseID number,
       journalisteID number,
       constraint pk_article primary key (articleID),
    constraint fk_article_presse foreign key (presseID) references
Presse(presseID),
       constraint fk_article_journaliste foreign key (journalisteID) references
Journaliste(journalisteID)
);


insert into Presse values (0001, NULL);
insert into Journaliste values (1001, 'Obama', 'Barack', NULL, NULL, 0001);
insert into Journaliste values (1002, 'Anonyme', 'Anonyme', NULL, 'T', 0001);
insert into Article values (2001, 'Article respectant la DTD1 du TP1 de
représentation des données du web.', 'DTD1', 0001, 1002);
insert into Article values (2002, 'Vous saurez tous sur comment j ai réagit quand j
ai appris que j avais reçu le prix Nobel de la paix', 'Moi et mon prix Nobel de la
paix', 0001, 1001);
```

3.

```
SELECT titre, corps, a.journalisteID, j.nom
FROM Article a, Journaliste j
WHERE a.journalisteID = j.journalisteID
AND j.anonyme = 'T';

SELECT titre, corps, j.nom, j.prenom
FROM Article a, Journaliste j
WHERE a.journalisteID = j.journalisteID
AND j.nom = 'Obama';

SELECT nom, prenom, anonyme, journalisteID
FROM Journaliste;
```

4.

```
DEFINE XML CONSTRUCT ARTICLE
(title : varchar (100),
body : varchar(1024),
idJ : number) as
{<article titre="$title"
auteur="$idJ">
       <corps>
              $body
       </corps>
</article>
}
```

```
DEFINE XML CONSTRUCT JOURNALISTE
(idJ number,
lstname varchar(50),
fstname varchar(50),
pseudo varchar(50)) as
{<journaliste idJ="$idJ">
       <anonyme/>
       <nom>$lstname</nom>
       <prenom>$fstname</nom>

<pseudonyme>$pseudo</pseudonyme>
</journaliste>
}
```

```
DEFINE XML CONSTRUCT
JOURNALISTES
(journaliste xml) as
{<journalistes>
       $journaliste
</journalistes>
}
```

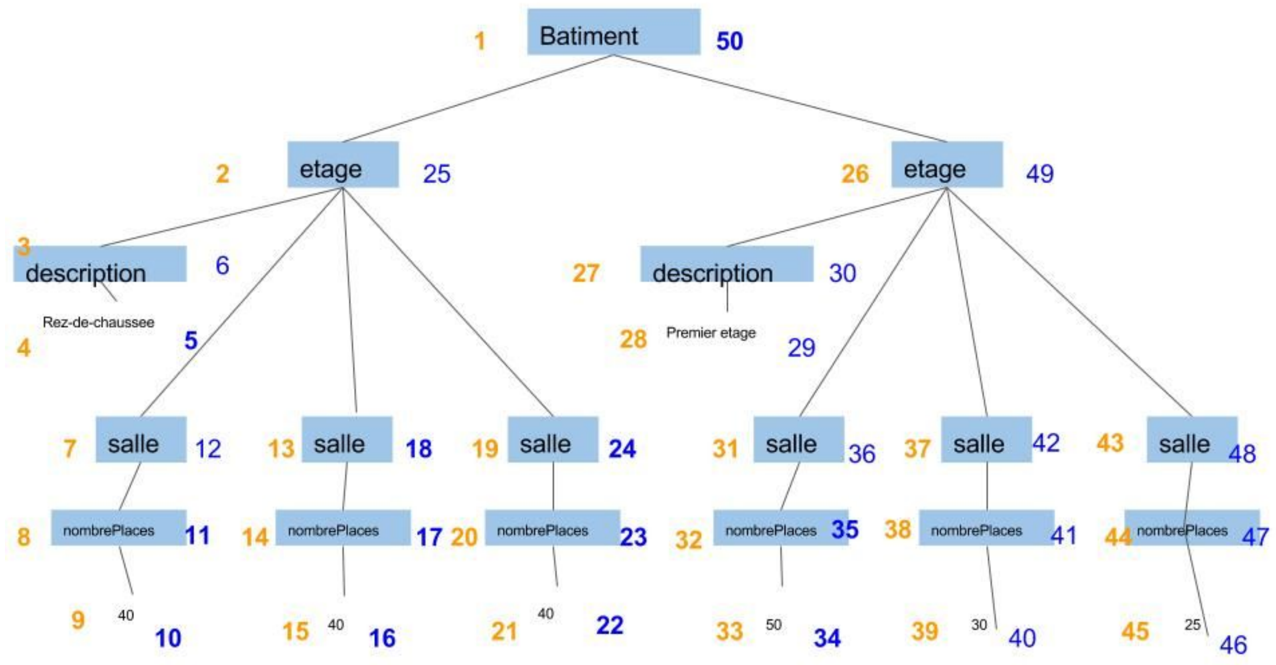
```
DEFINE XML CONSTRUCT
JOURNAL
(article xml) as
{<journal>
       $article
</journal>
}
```

```
DEFINE XML CONSTRUCT
PRESSE
(journal xml) as
{<presse>
       $journal
       $journalistes
</presse>
}
```

```
SELECT XMLAGG(SELECT presseID
       PRESSE((SELECT XMLAGG(
              JOURNAL((SELECT XMLAGG(
                     ARTICLE(SELECT titre, coprs, journalisteID
                                   FROM Article))
                            )),
              (SELECT XMLAGG(
              JOURNALISTES((SELECT XMLAGG(
                     JOURNALISTE(SELECT journalisteID, nom, prenom, anonyme,
pseudonyme))))))
              )))
       FROM Presse
       ORDER BY presseID);
```

4) Interval coding avec SAX

1 Encodage begin end de l’arbre batiment.



```
creationNode.sql :

create table node(
      id varchar(5),
      begin number(5),
      end number(5),
      parent varchar(5),
      tag varchar(20),
      nodetyp varchar(7),
      constraint pk_node primary key (id),
      constraint fk_node_node foreign key (parent) references node(id),
      constraint ck_node_nodetype check (nodetyp in ('element', 'text',
'number'))
);


create table textvalues(
      node varchar(5),
      value varchar(100),
      constraint pk_textvalues primary key (node),
      constraint fk_textvalues_node foreign key (node) references node(id)
);
```

```
create table numvalues(
      node varchar(5),
      value number(20),
      constraint pk_numvalues primary key (node),
      constraint fk_numvalues_node foreign key (node) references node(id)
);

create table attributes(
      node varchar(5),
      name varchar(30),
      value varchar(30),
      constraint pk_attributes primary key (node, name),
      constraint fk_attributes_node foreign key (node) references node(id)
);
```

2. Programmation encodage begin/end classe SaxeParser.java :

```
//Attributes :
private Hashtable<Integer, String> tags; //permet de trouver le tag d'un noeud grace a son
begin
private Hashtable<Integer, Integer> ids; //permet de trouver l'id d'un noeud grâce à sa
valeur begin
private int num;                                    //à incrémenter et associer au begin
d'un noeud
private Pattern patternD;                       //pour matcher des numvalues
private Pattern patternW;                      //pour matcher des textvalues
private int id;                             //à incrémeneter et associer à l'id d'un noeud
private Hashtable<Integer, String> reponses;        //pour mettre les requetes dans le bon
ordre
                                            //et eviter les références clé étrangère non
                                            //existante, on stocke les requetes
private Vector<String> attributes;          //stocke les requetes associées à la
                                            //table attributes


public void startDocument() throws SAXException {
       this.tags = new Hashtable<Integer, String>();
       this.num = 1;
       this.patternD = Pattern.compile("[0-9]+");
       this.patternW = Pattern.compile(".*[a-zA-Z-_].*");
       this.ids = new Hashtable<Integer, Integer>();
       this.id = 0;
       this.reponses = new Hashtable<Integer, String>();
       this.attributes = new Vector<String>();
}

public void startElement(String namespaceURI, String localName,
                              String qName, Attributes atts)
       throws SAXException
       {
       this.tags.put((Integer)this.num, localName);
       this.ids.put((Integer)this.num, (Integer)this.id);

       for(int i =0; i<atts.getLength(); i++){
```

```
                this.attributes.add("insert into attributes(node, name, value) values(
'n"+this.id+"', '"+atts.getLocalName(i)+"', '"+atts.getValue(i)+"');");
        }
        this.id++;
        this.num++;
}

public void endElement(String namespaceURI, String localName,
                                String qName)
        throws SAXException
        {
        Enumeration e = this.tags.keys();
        int beginParent = -1;
        int beginFils = -1;
        int val = 0;
        while(e.hasMoreElements()){
                val = (int) e.nextElement();
                if(val>beginFils){
                                beginParent = beginFils;
                                beginFils = val;
                }
                else if(val>beginParent){
                                beginParent = val;
                }
        }

        if(this.tags.get(beginFils).equals(localName)){
                this.reponses.put((Integer)this.ids.get(beginFils), "insert into node(id,
begin, end, parent, tag, nodetyp) values( 'n"+this.ids.get(beginFils)+"', "+beginFils+",
"+this.num+", "+(beginParent==-1?"NULL":"'n"+this.ids.get(beginParent)+"'")+",
'"+localName+"', 'element');");
                this.num++;
                this.tags.remove(beginFils);
        }
}

public void endDocument() throws SAXException {
        //on affiche les requetes dans l'ordre
        //on insere les noeuds depuis la valeur de begin la plus petite vers la plus grande
        for(int i =0; i<this.reponses.size(); i++){
                System.out.println(this.reponses.get(i));
        }
        //on affiche les requetes pour les attributs
        for(int i =0; i<this.attributes.size(); i++){
                System.out.println(this.attributes.get(i));
        }
}

public void characters(char[] ch, int start, int length) throws SAXException
        {

                String str = new String(ch, start, length);
                Matcher matcherD = this.patternD.matcher(str);
                Matcher matcherW = this.patternW.matcher(str);
                if(matcherD.matches()){
                int beginParent = 0;
                int val = 0;
                Enumeration e = this.tags.keys();
                while(e.hasMoreElements()){
                                val = (int) e.nextElement();
                                beginParent = val>beginParent?val:beginParent;
                }

                this.reponses.put((Integer)this.id, "insert into node(id, begin, end,
parent, tag, nodetyp) values( 'n"+this.id+"', "+this.num+", "+(this.num+1)+",
'n"+this.ids.get(beginParent)+"', NULL, 'number');\ninsert into numvalues(node, value)
values( 'n"+this.id+"', "+str+");");
                this.num+=2;
                this.id++;
                }
                else if(matcherW.matches()){
                int beginParent = 0;
                int val = 0;
                Enumeration e = this.tags.keys();
                while(e.hasMoreElements()){
                                val = (int) e.nextElement();
                                beginParent = val>beginParent?val:beginParent;
                }

                this.reponses.put((Integer)this.id, "insert into node(id, begin, end,
parent, tag, nodetyp) values( 'n"+this.id+"', "+this.num+", "+(this.num+1)+",
'n"+this.ids.get(beginParent)+"', NULL, 'text');\ninsert into textvalues(node, value)
values( 'n"+this.id+"', '"+str+"');");
                this.num+=2;
                this.id++;
                }
}
```

3. presse.xml :

```
<?xml version="1.0"?>
<presse>
    <journal>
        <article titre="DTD1" auteur="auth1">
                <corps>Article respectant la DTD1 du TP1 de
représentation des données du web.</corps>
        </article>
    </journal>
    <journalistes>
        <journaliste idJ="auth1">
                <anonyme></anonyme>
        </journaliste>
        <journaliste idJ="auth2">
                <nom>Obama</nom>
                <prenom>Barack</prenom>
        </journaliste>
    </journalistes>
</presse>
```

requetesNode.sql :

```
--//article [@auteur = //journaliste[child::anonyme]/@idJ]
select a1.value
from attributes a1,
attributes a2,
node n1
where a1.name='titre'
and a1.node = n1.id
and n1.tag= 'article'
and a2.node = n1.id
and a2.name = 'auteur'
and a2.value in (
      select a3.value
      from attributes a3,
      node n2,
      node n3
      where a3.name='idJ'
      and a3.node = n2.id
      and n3.parent = n2.id
      and n3.tag='anonyme'
);

--//article [position() = 1]/corps
select t1.value
from textvalues t1,
node n1,
node n2,
node n3
where t1.node=n1.id
and n1.nodetyp='text'
and n1.parent=n2.id
and n2.tag='corps'
and n2.parent = n3.id
and n3.tag = 'article'
and n3.begin in (
      select min(n4.begin)
      from node n4
      where n4.tag='article'
);

select a.value
from attributes a,
node n1,
node n2
where a.node=n1.id
and a.name='idJ'
and n1.tag='journaliste'
and n2.tag='nom'
and n2.parent=n1.id;
```

4. Test fichier XML standard

```
time java SaxParser standard
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
    at java.util.regex.Matcher.<init>(Matcher.java:225)
    at java.util.regex.Pattern.matcher(Pattern.java:1093)
    at SaxParser.characters(SaxParser.java:161)
    at
com.sun.org.apache.xerces.internal.parsers.AbstractSAXParser.characters(AbstractSAXParser.j
ava:546)
    at
com.sun.org.apache.xerces.internal.impl.XMLDocumentFragmentScannerImpl.scanDocument(XMLDocu
mentFragmentScannerImpl.java:454)
    at
com.sun.org.apache.xerces.internal.parsers.XML11Configuration.parse(XML11Configuration.java
:848)
    at
com.sun.org.apache.xerces.internal.parsers.XML11Configuration.parse(XML11Configuration.java
:777)
    at com.sun.org.apache.xerces.internal.parsers.XMLParser.parse(XMLParser.java:141)
    at
com.sun.org.apache.xerces.internal.parsers.AbstractSAXParser.parse(AbstractSAXParser.java:1
140)
    at
com.sun.org.apache.xerces.internal.jaxp.SAXParserImpl$JAXPSAXParser.parse(SAXParserImpl.jav
a:655)
    at SaxParser.main(SaxParser.java:248)

real    2m17.915s
user    2m26.744s
sys    0m1.172s
```